UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/020,420	12/13/2001	Vidyasagar Edara	INS-120	9550

35273 7590 03/25/2008
BEVER, HOFFMAN & HARMS, LLP
2099 GATEWAY PLACE
SUITE 320
SAN JOSE, CA 95110

EXAMINER	
CHEN, QING	
ART UNIT	**PAPER NUMBER**
2191	

MAIL DATE	DELIVERY MODE
03/25/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/020,420	EDARA ET AL.	
	Examiner	Art Unit	
	Qing Chen	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 December 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-83* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-83* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *13 December 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

1. This is the initial Office action based on the application filed on December 13, 2001.

2. **Claims 1-83** are pending.

Claim Objections

3. **Claims 4, 13, and 70** are objected to because of the following informalities:

- **Claim 4** contains a typographical error: "comprising of the step of" should read -- comprising the step of --.
- **Claim 13** recites the limitation "the address read from the processing component." Applicant is advised to change this limitation to read "the address received from the processing component" for the purpose of keeping the claim language consistent throughout the claims.
- **Claim 70** contains a typographical error: "The system of 68" should read -- The system of claim 68 --.

Appropriate correction is required.

Claim Rejections - 35 USC § 112

4. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

5. **Claims 3 and 4** are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 3 and 4 recite the limitation "the interpretive language instruction stream." There is insufficient antecedent basis for this limitation in the claims. In the interest of compact prosecution, the Examiner subsequently interprets this limitation as reading "an interpretive language instruction stream" for the purpose of further examination.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. **Claims 1, 9-16, 22-35, 37-39, 47-55, 61-73, and 75-77** are rejected under 35 U.S.C. 102(b) as being anticipated by **US 5,179,734 (hereinafter "Candy")**.

As per **Claim 1**, Candy discloses:

- coupling a hardware component with the processing component and the memory component *(see Column 4: 6-18, "The TIL processor includes a main input/output (I/O) data bus 10 and an address bus 12 which are interfaced with a I/O Subsystem and External Memory*

(Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.");

- employing the hardware component to assist processing of the interpretive language *(see Column 4: 6-18, "Referring now to FIG. 1, there is illustrated a basic block diagram of a threaded interpretive language (TIL) processor in accordance with the present invention." and "AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.")*; and

- permitting the system to execute the native software processes of the processing component *(see Column 4: 6-18, "The TIL processor of the present invention is capable of performing multiple data selection and arithmetic operations within a single or very few clock cycles.").*

As per **Claim 9**, the rejection of **Claim 1** is incorporated; and Candy further discloses:

- comparing at the hardware component an address received from the processing component to a fixed instruction fetch address stored at a decoding component of the hardware component *(see Column 7: 62-67 to Column 8: 1, "To appropriately address either a primitive control word or a functional control word in the microcode ROM 60, the address decoder 62 receives an address from the interface bus 64 and determines whether the address is a microcode address that corresponds to a primitive control word or an address that corresponds to address locations in either the memories 24 and 30.").*

As per **Claim 10**, the rejection of **Claim 9** is incorporated; and Candy further discloses:

- determining at the decoding component of the hardware component that the address received from the processing component does not correspond to the fixed instruction fetch address *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.")*, and

- sending the address received from the processing component to the memory component *(see Column 8: 1-7, "For example, addresses from 65 to 512 input to the address decoder 62 cause the address decoder 62 to address one address location in the microcode ROM 60 and output a functional control word.").*

As per **Claim 11**, the rejection of **Claim 9** is incorporated; and Candy further discloses:

- determining at the decoding component of the hardware component that the address received from the processing component corresponds to the fixed instruction fetch address stored at the decoding component *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 12**, the rejection of **Claim 10** is incorporated; and Candy further discloses:

- maintaining a current interpretive language address in an interpreter language program counter of the hardware component *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

As per **Claim 13**, the rejection of **Claim 12** is incorporated; and Candy further discloses:

- loading the interpreter language program counter with a starting address of an interpretive language instruction stream *(see Column 9: 61-65, "To determine whether the address should be loaded in the instruction pointer 86 or merely incremented and the previous address output therefrom, a decoder 92 is provided that is interfaced with the I/O bus 10.")*, and

- automatically incrementing the interpreter language program counter to the next address of the interpretive language instruction stream upon the address read from the processing component corresponding to the fixed instruction fetch address *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.")*.

As per **Claim 14**, the rejection of **Claim 12** is incorporated; and Candy further discloses:

- transmitting a select signal from the decoding component to an address multiplexer component of the hardware component *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).")*.

As per **Claim 15**, the rejection of **Claim 14** is incorporated; and Candy further discloses:

- transmitting the current interpretive language address from the interpreter language program counter of the hardware component to the memory component via the address multiplexer component based on the select signal received at the address multiplexer component from the decoding component *(see Column 21: 14-20, "A multiplexer 264 has two inputs that interface with the incrementing register 94 and the transfer gate 98 for interfacing with the I/O bus 10. The multiplexer 264 has a pair of outputs that interface with the return stack 20 through output interface buses 266 and a pair of outputs that interface with the shadow return stack 262 through output interface buses 268.")*.

As per **Claim 16**, the rejection of **Claim 15** is incorporated; and Candy further discloses:

- fetching data from a location at the memory component which is associated with the current interpretive language address *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 22**, the rejection of **Claim 1** is incorporated; and Candy further discloses:

- fetching from the memory component at least one operand of an interpretive language instruction stream *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 23**, the rejection of **Claim 22** is incorporated; and Candy further discloses:

- maintaining operand addresses at the hardware component *(see Column 4: 19-24, "A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses.").*

As per **Claim 24**, the rejection of **Claim 22** is incorporated; and Candy further discloses:

- in which the operand addresses are maintained at an interpreter language program counter of the hardware component *(see Column 9: 56-61, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

As per **Claim 25**, the rejection of **Claim 22** is incorporated; and Candy further discloses:

- fetching data which comprises at least one operand from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*, and

- storing the data comprising at least one operand at the hardware component *(see Column 4: 19-24, "A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out* *stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses.").*

As per **Claim 26**, the rejection of **Claim 25** is incorporated; and Candy further discloses:

- sending the data from the memory component via a data bus capable of carrying more than one operand to the hardware component *(see Column 4: 6-18, "The TIL processor includes a main input/output (I/O) data bus 10 and an address bus 12 which are interfaced with a I/O Subsystem and External Memory (Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.").*

As per **Claim 27**, the rejection of **Claim 26** is incorporated; and Candy further discloses:

- in which the data comprises at least one of: (a) an 8-bit operand, (b) a 16-bit operand, and (c) a 24-bit operand *(see Column 13: 23-26, "The RAM 154 provides both a Read and Write capability and is addressable through an 8-Bit address input to read or write data out of a 16-Bit data port.").*

As per **Claim 28**, the rejection of **Claim 25** is incorporated; and Candy further discloses:

- determining at the hardware component an operand size requested to be read by the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of* *addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*, and

- transmitting the operand requested by the processing component from the hardware component to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 29**, the rejection of **Claim 28** is incorporated; and <u>Candy</u> further discloses:

- determining at the hardware component if an address received from the processing component corresponds to a fixed operand fetch address stored at the hardware component *(see Column 7: 62-67 to Column 8: 1, "To appropriately address either a primitive control word or a functional control word in the microcode ROM 60, the address decoder 62 receives an address from the interface bus 64 and determines whether the address is a microcode address that corresponds to a primitive control word or an address that corresponds to address locations in either the memories 24 and 30.").*

As per **Claim 30**, the rejection of **Claim 22** is incorporated; and <u>Candy</u> further discloses:

- comparing at a decoding component of the hardware component an address received from the processing component to a number of fixed operand fetch address stored at the decoding component *(see Column 8: 1-7, "For addresses over 64, the address decoder 62* *addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 31**, the rejection of **Claim 30** is incorporated; and Candy further discloses:

- determining at the decoding component that the address is equal to a fixed operand fetch address *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 32**, the rejection of **Claim 30** is incorporated; and Candy further discloses:

- transmitting an operand address from an interpreter language program counter of the hardware component to the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 33**, the rejection of **Claim 31** is incorporated; and Candy further discloses:

- transmitting a select signal from the decoding component to an address multiplexer component of the hardware component *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 34**, the rejection of **Claim 33** is incorporated; and Candy further discloses:

- fetching data associated with the operand address from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 35**, the rejection of **Claim 34** is incorporated; and Candy further discloses:

- transmitting the data from the memory component via a data bus to an operand storing component of the hardware component *(see Column 4: 6-18, "The TIL processor includes a main input/output (I/O) data bus 10 and an address bus 12 which are interfaced with a I/O Subsystem and External Memory (Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.")*, and

- wherein the data comprises at least one operand of the interpretive language instruction stream *(see Column 4: 19-24, "A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses.").*

As per **Claim 37**, the rejection of **Claim 35** is incorporated; and Candy further discloses:

- determining at the decoding component an operand size requested to be read by the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*, and

- transmitting the operand of the size requested by the processing component from the operand storing component to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 38**, the rejection of **Claim 37** is incorporated; and Candy further discloses:

- transmitting a data multiplexer select signal to a data multiplexer component of the hardware component in response to the decoding component determining that the address received from the processing component corresponds to the fixed operand fetch address *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).")*, and

- transmitting the operand to the processing component via the data multiplexer component based on the data multiplexer select signal received from the decoder component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus*

12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").

As per **Claim 39**, the rejection of **Claim 38** is incorporated; and Candy further discloses:

- storing multiple bytes at the operand storing component *(see Column 4: 19-24, "A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses."),* and

- shifting the multiple operand bytes at the operand storing component into an order required by the processing component *(see Column 4: 30-33, "These microcode instructions determined how the data is shifted around in the stacks 18 and 20 and what functions the ALU 16 performs on data input thereto.").*

As per **Claim 47**, Candy discloses:

- a hardware component coupled with the processing component and the memory component to assist processing of the interpretive language with the system able to execute the native software processes of the processing component *(see Column 4: 6-18, "Referring now to FIG. 1, there is illustrated a basic block diagram of a threaded interpretive language (TIL) processor in accordance with the present invention. The TIL processor of the present invention is capable of performing multiple data selection and arithmetic operations within a single or very few clock cycles. The TIL processor includes a main input/output (I/O) data bus 10 and an* *address bus 12 which are interfaced with a I/O Subsystem and External Memory (Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.").*

As per **Claim 48**, the rejection of **Claim 47** is incorporated; and <u>Candy</u> further discloses:

- a decoding component of the hardware component that stores a fixed instruction fetch address, said decoding component compares an address received from the processing component with the fixed instruction fetch address *(see Column 7: 62-67 to Column 8: 1, "To appropriately address either a primitive control word or a functional control word in the microcode ROM 60, the address decoder 62 receives an address from the interface bus 64 and determines whether the address is a microcode address that corresponds to a primitive control word or an address that corresponds to address locations in either the memories 24 and 30.").*

As per **Claim 49**, the rejection of **Claim 48** is incorporated; and <u>Candy</u> further discloses:

- in which the decoding component determines if the address received from the processing component corresponds to the fixed instruction fetch address *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 50**, the rejection of **Claim 49** is incorporated; and <u>Candy</u> further discloses:

- in which the address received from the processing component is sent to the memory component via the hardware component in response to the decoding component determining that the address received does not correspond to the fixed instruction fetch address *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses. For example, addresses from 65 to 512 input to the address decoder 62 cause the address decoder 62 to address one address location in the microcode ROM 60 and output a functional control word.")*.

As per **Claim 51,** the rejection of **Claim 49** is incorporated; and Candy further discloses:

- an interpreter language program counter of the hardware component which maintains a current interpretive language address *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.")*.

As per **Claim 52,** the rejection of **Claim 51** is incorporated; and Candy further discloses:

- in which the interpreter language program counter is loaded with a starting address of an interpretive language instruction stream, said interpreter language program counter is automatically incremented to the next address of the interpretive language instruction stream upon the address read from the processing component corresponding to the fixed instruction fetch address *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein." and 61-65, "To determine* *whether the address should be loaded in the instruction pointer 86 or merely incremented and the previous address output therefrom, a decoder 92 is provided that is interfaced with the I/O bus 10.").*

As per **Claim 53**, the rejection of **Claim 51** is incorporated; and Candy further discloses:

- an address multiplexer component of the hardware component which receives a select signal transmitted from the decoding component *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 54**, the rejection of **Claim 53** is incorporated; and Candy further discloses:

- in which the interpreter language program counter transmits the current interpretive language address to the memory component via the address multiplexer component based on the select signal received at the address multiplexer component from the decoding component *(see Column 21: 14-20, "A multiplexer 264 has two inputs that interface with the incrementing register 94 and the transfer gate 98 for interfacing with the I/O bus 10. The multiplexer 264 has a pair of outputs that interface with the return stack 20 through output interface buses 266 and a pair of outputs that interface with the shadow return stack 262 through output interface buses 268.").*

As per **Claim 55**, the rejection of **Claim 54** is incorporated; and Candy further discloses:

- in which data is fetched from a location at the memory component which is associated with the current interpretive language address *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 61**, the rejection of **Claim 47** is incorporated; and Candy further discloses:

- in which at least one operand of an interpretive language instruction stream is fetched from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 62**, the rejection of **Claim 61** is incorporated; and Candy further discloses:

- an interpreter language program counter of the hardware component for maintaining operand addresses *(see Column 9: 56-61, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

As per **Claim 63**, the rejection of **Claim 61** is incorporated; and Candy further discloses:

- in which the hardware component stores data which comprises at least one operand which is fetched from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 64**, the rejection of **Claim 63** is incorporated; and Candy further discloses:

- a data bus coupled with the memory component and the hardware component which is capable of carrying more than one operand of data sent from the memory component to the hardware component *(see Column 4: 6-18, "The TIL processor includes a main input/output (I/O) data bus 10 and an address bus 12 which are interfaced with a I/O Subsystem and External Memory (Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto.").*

As per **Claim 65**, the rejection of **Claim 64** is incorporated; and Candy further discloses:

- in which the data comprises at least one of: (a) an 8-bit operand, (b) a 16-bit operand, and (c) a 24-bit operand *(see Column 13: 23-26, "The RAM 154 provides both a Read and Write capability and is addressable through an 8-Bit address input to read or write data out of a 16-Bit data port.").*

As per **Claim 66**, the rejection of **Claim 64** is incorporated; and Candy further discloses:

- in which the hardware component determines an operand size requested to be read by the processing component and the hardware component transmits the operand of the size requested by the processing component to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 67**, the rejection of **Claim 66** is incorporated; and <u>Candy</u> further discloses:

- in which the hardware component determines if an address received from the processing component corresponds to one of a number of fixed operand fetch address stored at the hardware component *(see Column 7: 62-67 to Column 8: 1, "To appropriately address either a primitive control word or a functional control word in the microcode ROM 60, the address decoder 62 receives an address from the interface bus 64 and determines whether the address is a microcode address that corresponds to a primitive control word or an address that corresponds to address locations in either the memories 24 and 30.").*

As per **Claim 68**, the rejection of **Claim 61** is incorporated; and <u>Candy</u> further discloses:

- a decoding component of the hardware component which compares an address received from the processing component to the fixed operand fetch addresses stored at the decoding component *(see Column 8: 1-7, "For addresses over 64, the address decoder 62* *addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 69**, the rejection of **Claim 68** is incorporated; and Candy further discloses:

- in which the decoding component determines that the address corresponds to one of the fixed operand fetch addresses *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.").*

As per **Claim 70**, the rejection of **Claim 68** is incorporated; and Candy further discloses:

- an interpreter language program counter of the hardware component which transmits an operand address to the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 71**, the rejection of **Claim 69** is incorporated; and Candy further discloses:

- in which the decoding component transmits a select signal to an address multiplexer of the hardware component *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 72**, the rejection of **Claim 71** is incorporated; and Candy further discloses:

- in which data associated with the operand address is fetched from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 73**, the rejection of **Claim 72** is incorporated; and Candy further discloses:

- an operand storing component of the hardware component which receives data transmitted from the memory component via a data bus to the operand storing component wherein the data comprises at least one operand of the interpretive language instruction stream *(see Column 4: 6-24, "The TIL processor includes a main input/output (I/O) data bus 10 and an address bus 12 which are interfaced with a I/O Subsystem and External Memory (Memory) 14. AN arithmetic logic unit (ALU) 16 is interfaced with the I/O bus 10 and is operable to perform arithmetic and logic functions on data input thereto. A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses.").*

As per **Claim 75**, the rejection of **Claim 73** is incorporated; and Candy further discloses:

- in which the decoding component determines an operand size requested to be read by the processing component and the operand storing component transmits the operand of the size requested to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 76**, the rejection of **Claim 75** is incorporated; and Candy further discloses:

- a data multiplexer component of the hardware component which receives a data multiplexer select signal in response to the decoding component determining that the address received from the processing component is equal to a fixed operand fetch address, said operand is transmitted to the processing component via the data multiplexer component based on the data multiplexer select signal received from the decoder component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction."; Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).")*.

As per **Claim 77**, the rejection of **Claim 76** is incorporated; and Candy further discloses:

- in which the operand storing component stores multiple operands and the operand storing component shifts the multiple operands into an order required by the processing component *(see Column 4: 30-33, "These microcode instructions determined how the data is* *shifted around in the stacks 18 and 20 and what functions the ALU 16 performs on data input thereto.").*

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. **Claims 2-8, 17-21, 40-42, 56-60, and 78-80** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Candy** in view of **US 6,658,655 (hereinafter "Hoogerbrugge")**.

As per **Claim 2**, the rejection of **Claim 1** is incorporated; however, <u>Candy</u> does not disclose:

- generating an instruction jump address at the hardware component.

<u>Hoogerbrugge</u> discloses:

- generating an instruction jump address at the hardware component *(see Column 5: 38-52, "However, in a block of the threaded interpreter the following elements can be distinguished: ... a jump instruction to the block corresponding to the type of program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of <u>Hoogerbrugge</u> into the teaching of <u>Candy</u> to include generating an instruction jump address at the hardware component. The modification would be obvious because one of ordinary skill in the art would be motivated to jump to the next program instruction to be executed *(see Hoogerbrugge – Column 2: 37-39).*

As per **Claim 3**, the rejection of **Claim 2** is incorporated; and Candy further discloses:

- maintaining a current address for the interpretive language instruction stream at the hardware component *(see Column 4: 19-24, "A parameter stack 18 is interfaced with the ALU 16 and is a last-in first-out stack implemented in hardware. A return stack 20, which is also a last-in first-out stack, is interfaced with the I/O bus 10 and is operable for storing data and addresses.").*

As per **Claim 4**, the rejection of **Claim 3** is incorporated; and Candy further discloses:

- automatically incrementing the current address for the interpretive language instruction stream at the hardware component in response to an address sent from the processing component and received at the hardware component corresponding to a fixed instruction fetch address stored at the hardware component *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

As per **Claim 5**, the rejection of **Claim 2** is incorporated; and Candy further discloses:

- in which the generating of the instruction jump address is in response to the hardware component determining that an address received from the processing component corresponds to a fixed instruction fetch address stored at the hardware component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 6**, the rejection of **Claim 5** is incorporated; and Candy further discloses:

- transmitting a current interpretive language address from the hardware component to the memory component upon the hardware component determining that the address received from the processing component corresponds to the fixed instruction fetch address *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 7**, the rejection of **Claim 6** is incorporated; and Candy further discloses:

- transmitting the instruction jump address from the hardware component to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of* *addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 8**, the rejection of **Claim 7** is incorporated; and Candy further discloses:

- fetching data associated with the current interpretive language address from the memory component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*,

- transmitting the data from the memory component to the hardware component wherein the data comprises at least a current interpretive instruction *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*, and

- shifting the current interpretive instruction by a predetermined number of bits at the hardware component to establish a shifted address *(see Column 4: 30-33, "These microcode instructions determined how the data is shifted around in the stacks 18 and 20 and what functions the ALU 16 performs on data input thereto.").*

However, Candy does not disclose:

- adding the shifted address to a base address stored in the hardware component to calculate the instruction jump address.

Hoogerbrugge discloses:

- adding the shifted address to a base address stored in the hardware component to calculate the instruction jump address *(see Column 11: 25-30, "The switch of basic block 702 is moved backwards in the loop and, together with the jump of basic block 712, added to each of the basic blocks 704 to 710. The switch plus the jump represent the decode of a program instruction and the jump to the block of that program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include adding the shifted address to a base address stored in the hardware component to calculate the instruction jump address. The modification would be obvious because one of ordinary skill in the art would be motivated to optimize the calculation of the instruction jump address.

As per **Claim 17**, the rejection of **Claim 16** is incorporated; however, Candy does not disclose:

- transmitting the data from the memory component to an instruction jump address generator component of the hardware component, wherein the data comprises at least a current interpretive instruction.

Hoogerbrugge discloses:

- transmitting the data from the memory component to an instruction jump address generator component of the hardware component, wherein the data comprises at least a current interpretive instruction *(see Column 5: 38-52, "However, in a block of the threaded interpreter* *the following elements can be distinguished: ... a jump instruction to the block corresponding to the type of program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include transmitting the data from the memory component to an instruction jump address generator component of the hardware component, wherein the data comprises at least a current interpretive instruction. The modification would be obvious because one of ordinary skill in the art would be motivated to jump to the next program instruction to be executed *(see Hoogerbrugge – Column 2: 37-39).*

As per **Claim 18**, the rejection of **Claim 17** is incorporated; and Candy further discloses:

- shifting the current interpretive instruction by a predetermined number of bits at the instruction jump address generator component to establish a shifted address *(see Column 4: 30-33, "These microcode instructions determined how the data is shifted around in the stacks 18 and 20 and what functions the ALU 16 performs on data input thereto.").*

However, Candy does not disclose:

- adding the shifted address to a base address stored in the instruction jump address generator component to calculate an instruction jump address.

Hoogerbrugge discloses:

- adding the shifted address to a base address stored in the instruction jump address generator component to calculate an instruction jump address *(see Column 11: 25-30, "The switch of basic block 702 is moved backwards in the loop and, together with the jump of basic* *block 712, added to each of the basic blocks 704 to 710. The switch plus the jump represent the decode of a program instruction and the jump to the block of that program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include adding the shifted address to a base address stored in the instruction jump address generator component to calculate an instruction jump address. The modification would be obvious because one of ordinary skill in the art would be motivated to optimize the calculation of the instruction jump address.

As per **Claim 19**, the rejection of **Claim 18** is incorporated; and Candy further discloses:

- transmitting the instruction jump address from the instruction jump address generator component of the hardware component to a data multiplexer component of the hardware component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 20**, the rejection of **Claim 19** is incorporated; and Candy further discloses:

- transmitting a data multiplexer select signal from the decoding component to the data multiplexer component of the hardware component based on the address received at the decoding component from the processing component corresponding to the fixed instruction fetch address *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to* *switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 21**, the rejection of **Claim 20** is incorporated; and Candy further discloses:

- transmitting the instruction jump address from the data multiplexer component of the hardware component to the processing component based on the data multiplexer select signal from the decoding component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 40**, the rejection of **Claim 1** is incorporated; however, Candy does not disclose:

- storing a thread switch jump address at the hardware component.

Hoogerbrugge discloses:

- storing a thread switch jump address at the hardware component *(see Column 5: 38-52, "However, in a block of the threaded interpreter the following elements can be distinguished: ... a jump instruction to the block corresponding to the type of program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include storing a thread switch jump address at the hardware component. The modification would be obvious because one of ordinary skill in the art would be motivated to jump to the next program instruction to be executed *(see* <u>*Hoogerbrugge*</u> *– Column 2: 37-39).*

As per **Claim 41**, the rejection of **Claim 40** is incorporated; and <u>Candy</u> further discloses:

- transmitting the thread switch jump address from the hardware component to processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 42**, the rejection of **Claim 41** is incorporated; and <u>Candy</u> further discloses:

- determining at the hardware component if an address received from the processing component is equal to a fixed instruction fetch address *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses."),* and

- incrementing a counter component of the hardware component in response to the determination that the address received from the processing component corresponds to the fixed instruction fetch address *(see Column 9: 56-41, "To initiate a threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

As per **Claim 56**, the rejection of **Claim 55** is incorporated; however, Candy does not disclose:

- an instruction jump address generator component of the hardware component which receives the data transmitted from the memory component, wherein the data comprises at least a current interpretive instruction.

Hoogerbrugge discloses:

- an instruction jump address generator component of the hardware component which receives the data transmitted from the memory component, wherein the data comprises at least a current interpretive instruction *(see Column 5: 38-52, "However, in a block of the threaded interpreter the following elements can be distinguished: ... a jump instruction to the block corresponding to the type of program instruction.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include an instruction jump address generator component of the hardware component which receives the data transmitted from the memory component, wherein the data comprises at least a current interpretive instruction. The modification would be obvious because one of ordinary skill in the art would be motivated to jump to the next program instruction to be executed *(see Hoogerbrugge – Column 2: 37-39)*.

As per **Claim 57**, the rejection of **Claim 56** is incorporated; and Candy further discloses:

- in which the instruction jump address generator component shifts the current interpretive instruction by a predetermined number of bits to establish a shifted address *(see*

Column 4: 30-33, "These microcode instructions determined how the data is shifted around in the stacks 18 and 20 and what functions the ALU 16 performs on data input thereto.").

However, Candy does not disclose:

- said instruction jump address generator component stores a base address which is added to the shifted address to calculate an instruction jump address.

Hoogerbrugge discloses:

- said instruction jump address generator component stores a base address which is added to the shifted address to calculate an instruction jump address *(see Column 11: 25-30, "The switch of basic block 702 is moved backwards in the loop and, together with the jump of basic block 712, added to each of the basic blocks 704 to 710. The switch plus the jump represent the decode of a program instruction and the jump to the block of that program instruction.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include said instruction jump address generator component stores a base address which is added to the shifted address to calculate an instruction jump address. The modification would be obvious because one of ordinary skill in the art would be motivated to optimize the calculation of the instruction jump address.

As per **Claim 58**, the rejection of **Claim 57** is incorporated; and Candy further discloses:

- a data multiplexer component of the hardware component which receives the instruction jump address transmitted from the instruction jump address generator component *(see*

Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").

As per **Claim 59**, the rejection of **Claim 58** is incorporated; and Candy further discloses:

- in which the decoding component transmits a data multiplexer select signal to the data multiplexer component based on the address received at the decoding component from the processing component corresponding to the fixed instruction fetch address *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 60**, the rejection of **Claim 59** is incorporated; and Candy further discloses:

- in which the instruction jump address is transmitted from the data multiplexer component to the processing component based on the data multiplexer select signal from the decoding component *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

As per **Claim 78**, the rejection of **Claim 47** is incorporated; however, Candy does not disclose:

- a thread switch jump address stored at the hardware component.

Hoogerbrugge discloses:

- a thread switch jump address stored at the hardware component *(see Column 5: 38-52, "However, in a block of the threaded interpreter the following elements can be distinguished: ... a jump instruction to the block corresponding to the type of program instruction.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Hoogerbrugge into the teaching of Candy to include a thread switch jump address stored at the hardware component. The modification would be obvious because one of ordinary skill in the art would be motivated to jump to the next program instruction to be executed *(see Hoogerbrugge – Column 2: 37-39)*.

As per **Claim 79**, the rejection of **Claim 78** is incorporated; and Candy further discloses:

- in which the hardware component transmits the thread switch jump address to the processing component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 80**, the rejection of **Claim 79** is incorporated; and Candy further discloses:

- a counter component of the hardware component which is incremented in response to a determination at the hardware component that an address received from the processing component corresponds to a fixed instruction fetch address *(see Column 9: 56-41, "To initiate a* *threaded list, the initial address of the threaded list is placed on the I/O bus 10 from either the ROM/RAM 80 or the memory 14 and loaded into the instruction register 58. This address is then clocked directly through the instruction pointer 86 to the address bus 12 and incremented therein.").*

10. **Claims 36 and 74** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Candy** in view of **US 6317872 (hereinafter "Gee")**.

As per **Claim 36**, the rejection of **Claim 35** is incorporated; however, Candy does not disclose:

- in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component.

Gee discloses:

- in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component *(see Column 8: 63-65, "Instruction bytecodes are fetched from code memory 104 four at a time over 32-bit data bus 216 and stored in the instruction register 266.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Gee into the teaching of Candy to include in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component. The modification would be obvious because one of ordinary skill in the art would be motivated to utilize the 32-bit computer architecture.

As per **Claim 74**, the rejection of **Claim 73** is incorporated; however, <u>Candy</u> does not disclose:

- in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component.

<u>Gee</u> discloses:

- in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component *(see Column 8: 63-65, "Instruction bytecodes are fetched from code memory 104 four at a time over 32-bit data bus 216 and stored in the instruction register 266.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of <u>Gee</u> into the teaching of <u>Candy</u> to include in which the data bus is a 32-bit data bus such that four bytes of 8-bit data are sent from the memory component to the operand storing component. The modification would be obvious because one of ordinary skill in the art would be motivated to utilize the 32-bit computer architecture.

11. **Claims 43-46 and 81-83** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Candy** in view of **Hoogerbrugge** as applied to Claims 42 and 80 above, and further in view of **US 5,586,256 (hereinafter "Thiel")**.

As per **Claim 43**, the rejection of **Claim 42** is incorporated; however, Candy and Hoogerbrugge do not disclose:

- counting a total number of addresses received from the processing component which are determined to correspond to the fixed instruction fetch address, and

- determining that the total number counted has reached a predetermined number.

Thiel discloses:

- counting a total number of addresses received from the processing component which are determined to correspond to the fixed instruction fetch address *(see Column 6: 54-56, "The loop countrer 66 records the total number of addresses generated and stops the process at the end.")*, and

- determining that the total number counted has reached a predetermined number *(see Column 6: 47-49, "If any of the count values becomes equal to its corresponding boundary value then the EQUAL signal for that dimension is asserted.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Thiel into the teaching of Candy to include counting a total number of addresses received from the processing component which are determined to correspond to the fixed instruction fetch address, and determining that the total number counted has reached a predetermined number. The modification would be obvious because one of ordinary skill in the art would be motivated to prevent instruction processing overflow.

As per **Claim 44**, the rejection of **Claim 43** is incorporated; and Candy further discloses:

- determining if an address received from the processing component is equal to the fixed instruction fetch address is performed at a decoding component of the hardware component *(see Column 8: 1-7, "For addresses over 64, the address decoder 62 addresses a particular address location in the microcode ROM 60 for a given group of addresses.")*.

As per **Claim 45**, the rejection of **Claim 44** is incorporated; and Candy further discloses:

- transmitting the thread switch jump address from the counter component to a data multiplexer component of the hardware component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 46**, the rejection of **Claim 45** is incorporated; and Candy further discloses:

- transmitting a data multiplexer select signal from the decoder component to the data multiplexer component in response to the total number counted at the counter component reaching the predetermined number *(see Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).")*, and

- transmitting the thread switch jump address from the counter component to the processing component via the data multiplexer component based on the data multiplexer select signal *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.").*

As per **Claim 81**, the rejection of **Claim 80** is incorporated; however, Candy and Hoogerbrugge do not disclose:

- in which the counter component counts a total number of addresses received from the processing component which are determined by a decoding component of the hardware component to correspond to the fixed instruction fetch address and the counter component determines that the total number counted has reached a predetermined number.

Thiel discloses:

- in which the counter component counts a total number of addresses received from the processing component which are determined by a decoding component of the hardware component to correspond to the fixed instruction fetch address and the counter component determines that the total number counted has reached a predetermined number *(see Column 6: 47-49, "If any of the count values becomes equal to its corresponding boundary value then the EQUAL signal for that dimension is asserted." and 54-56, "The loop countrer 66 records the total number of addresses generated and stops the process at the end.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Thiel into the teaching of Candy to include in which the counter component counts a total number of addresses received from the processing component which are determined by a decoding component of the hardware component to correspond to the fixed instruction fetch address and the counter component determines that the total number counted has reached a predetermined number. The modification would be obvious because one of ordinary skill in the art would be motivated to prevent instruction processing overflow.

As per **Claim 82**, the rejection of **Claim 81** is incorporated; and Candy further discloses:

- a data multiplexer component of the hardware component which receives the thread switch jump address transmitted from the counter component *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction.")*.

As per **Claim 83**, the rejection of **Claim 82** is incorporated; and Candy further discloses:

- in which a data multiplexer select signal is transmitted from the decoder component to the data multiplexer component in response to the total number counted at the counter component reaching a predetermined number and the thread switch jump address is transmitted from the counter component to the processing component via the data multiplexer component based on the data multiplexer select signal *(see Column 4: 53-58, "A first level memory 24 is interfaced with the I/O bus 10 and the address bus 12 and is operable to store microcode instructions in the form of addresses for the data manipulator 22. These instructions are received* *by the data manipulator 22 from the I/O bus 10 to thereby execute the instruction."; Column 20: 67 and 68 to Column 21: 1 and 2, "The multiplexer 248 is operable to switch the input interface buses 250 or 252 to interface with the output data buses 46 and 48. The multiplexers 242 and 248 are controlled by a multiplex signal (MUX).").*

Conclusion

12. The prior art made of record and not relied upon is considered pertinent to Applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Qing Chen whose telephone number is 571-270-1071. The Examiner can normally be reached on Monday through Thursday from 7:30 AM to 4:00 PM. The Examiner can also be reached on alternate Fridays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Wei Zhen, can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/QC/
March 3, 2008
/Wei Zhen/

Supervisory Patent Examiner, Art Unit 2191